

A

Please type a plus sign (+) inside this box → ☐PTO/SB/05 (4/98)
Approved for use through 09/30/2000. OMB 0651-0032
Patent and Trademark Office U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number**UTILITY
PATENT APPLICATION
TRANSMITTAL**

(Only for new nonprovisional applications under 37 C.F.R. § 1.53(b))

Attorney Docket No. TRI-003

First Inventor or Application Identifier Brian Fox

Title "Configuration In A Configurable System On A Chip"

Express Mail Label No. EL188845458US

APPLICATION ELEMENTS

See MPEP chapter 600 concerning utility patent application contents.

ADDRESS TO: Assistant Commissioner for Patents
Box Patent Application
Washington, DC 20231

1. ☐ * Fee Transmittal Form (e.g., PTO/SB/17)
(Submit an original and a duplicate for fee processing)
2. ☒ Specification [Total Pages 26]
(preferred arrangement set forth below)
- Descriptive title of the Invention
 - Cross References to Related Applications
 - Statement Regarding Fed sponsored R & D
 - Reference to Microfiche Appendix
 - Background of the Invention
 - Brief Summary of the Invention
 - Brief Description of the Drawings (if filed)
 - Detailed Description
 - Claim(s)
 - Abstract of the Disclosure
3. ☒ Drawing(s) (35 U.S.C. 113) [Total Sheets 11]
4. Oath or Declaration [Total Pages]
- a. ☐ Newly executed (original or copy)
 - b. ☐ Copy from a prior application (37 C.F.R. § 1.63(d))
(for continuation/divisional with Box 16 completed)
 - i. ☐ DELETION OF INVENTOR(S)
Signed statement attached deleting
inventor(s) named in the prior application,
see 37 C.F.R. §§ 1.63(d)(2) and 1.33(b).

5. ☐ Microfiche Computer Program (Appendix)
6. Nucleotide and/or Amino Acid Sequence Submission
(if applicable, all necessary)
- a. ☐ Computer Readable Copy
 - b. ☐ Paper Copy (identical to computer copy)
 - c. ☐ Statement verifying identity of above copies

ACCOMPANYING APPLICATION PARTS

7. ☐ Assignment Papers (cover sheet & document(s))
8. ☐ 37 C.F.R. § 3.73(b) Statement ☐ Power of Attorney
(when there is an assignee)
9. ☐ English Translation Document (if applicable)
10. ☒ Information Disclosure Statement (IDS)/PTO-1449 ☐ Copies of IDS Citations
11. ☐ Preliminary Amendment
12. ☒ Return Receipt Postcard (MPEP 503)
(Should be specifically itemized)
13. ☐ * Small Entity Statement filed in prior application
Statement(s) ☐ Status still proper and desired
(PTO/SB/09-12)
14. ☐ Certified Copy of Priority Document(s)
(if foreign priority is claimed)
15. ☐ Other:

* NOTE FOR ITEMS 1 & 13 IN ORDER TO BE ENTITLED TO PAY SMALL ENTITY
FEES, A SMALL ENTITY STATEMENT IS REQUIRED (37 C.F.R. § 1.27), EXCEPT
IF ONE FILED IN A PRIOR APPLICATION IS RELIED UPON (37 C.F.R. § 1.28).

16. If a CONTINUING APPLICATION, check appropriate box, and supply the requisite information below and in a preliminary amendment:

☐ Continuation ☐ Divisional ☐ Continuation-in-part (CIP) of prior application No: _____

Prior application information: Examiner _____ Group / Art Unit: _____

For CONTINUATION or DIVISIONAL APPS only: The entire disclosure of the prior application, from which an oath or declaration is supplied under Box 4b, is considered a part of the disclosure of the accompanying continuation or divisional application and is hereby incorporated by reference. The incorporation can only be relied upon when a portion has been inadvertently omitted from the submitted application parts.

17. CORRESPONDENCE ADDRESS☐ Customer Number or Bar Code Label

(Insert Customer No. or Attach bar code label here)

or ☒ Correspondence address below

Name	Jeanette S. Harms				
	BEVER & HOFFMAN, LLP				
Address	2099 Gateway Place, Suite 320				
City	San Jose	State	CA	Zip Code	95110-1017
Country		Telephone	(408) 451-5907	Fax	(408) 451-5908

Name (Print/Type)	Jeanette S. Harms	Registration No. (Attorney/Agent)	35,537
Signature			Date 10-15-99

Burden Hour Statement. This form is estimated to take 0.2 hours to complete. Time will vary depending upon the needs of the individual case. Any comments on the amount of time you are required to complete this form should be sent to the Chief Information Officer, Patent and Trademark Office, Washington, DC 20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Assistant Commissioner for Patents, Box Patent Application, Washington, DC 20231.

1 CONFIGURATION IN A CONFIGURABLE SYSTEM ON A CHIP

2 Brian Fox

3 Andreas Papaliolios

4
5 BACKGROUND OF THE INVENTION6 Field of the Invention

7 The present invention relates generally to a
8 configurable system on a chip (CSoC), and specifically to
9 structures and methods regarding the configuration of the
10 CSoC.

11
12 Description of the Related Art

13 Programmable logic devices, such as field programmable
14 logic devices (FPLDs), are programmed to perform user-
15 specified logic functions by loading configuration data
16 into the FPLD. This configuration data is typically loaded
17 into the FPLD as a bitstream, i.e. a string of binary bits.
18 Each bit programs a specific programmable resource on the
19 device. Thus, some bits configure the logic blocks which
20 perform the user-defined logic functions, other bits
21 configure the input/output blocks which interface to
22 devices external to the FPGA, and yet other bits configure
23 the programmable interconnect that connects the logic
24 blocks and the input/output blocks.

25 Typically, the configuration data is stored in a
26 nonvolatile memory device and loaded into the FPLD upon
27 device power-up. The configuration information is loaded
28 into the FPLD in data frames using a shift register. This
29 shift register has N serially coupled one-bit shift
30 registers, each register clocked by the same clock signal.
31 As the bitstream is serially shifted into the shift

1 register, the bits of the shift register are bit shifted
2 downstream in a synchronized manner.

3 Once the shift register is fully loaded, thereby
4 indicating a frame of data is complete, the stored N bits
5 are transferred simultaneously via dedicated lines to some
6 of the configuration memory cells. Typically, the
7 configuration memory cells are in an array, wherein a frame
8 of data corresponds to one column of configuration memory
9 cells in that array. After the write cycle is complete,
10 the loading of another frame of configuration data begins.
11 These store and write cycles continue until all of the bits
12 of the configuration bitstream are written into the FPLD.

13 Certain programming limitations are inherent in such
14 FPLDs. For example, because each bit corresponds to a
15 specific programmable resource and each frame of
16 configuration data is loaded in a fixed sequence,
17 reconfiguration of many FPLDs is an "all or nothing"
18 process. In other words, because there is no way to
19 restrict reconfiguration to a part of the device, the
20 entire bitstream must be reloaded into the FPLD. Moreover,
21 the shift register and dedicated lines take up valuable
22 silicon real estate.

23 Therefore, a need arises for a structure and method of
24 providing programmable logic solutions, while at the same
25 time ensuring more efficient use of silicon and system
26 resources.

27

28 SUMMARY OF THE INVENTION

29 The present invention allows a user to customize the
30 configuration sequence of a configurable system on a chip
31 (CSoC), thereby adding considerable flexibility to the
32 configuration process. The present invention also provides

1 certain features, transparent to the user, which optimize
2 system resources and ensure the correct initialization of
3 the CSoC.

4 In accordance with one aspect of the present
5 invention, the CSoC leverages an on-chip central processing
6 unit (CPU) to control the configuration process of the
7 configurable system logic (CSL). Advantageously, the CSL
8 configuration memory cells as well as other programmable
9 locations in the CSoC are addressable as part of a system
10 bus address space, thereby eliminating the dedicated
11 configuration shift register of prior art FPLDs.

12 Moreover, the present invention provides that this
13 system bus is a multi-use structure. Thus, the system bus
14 of the CSoC may be used for both configuring and reading of
15 memory cells. In this manner, the present invention
16 substantially eliminates the configuration interconnect
17 used in prior art FPLDs, thereby optimizing system
18 resources.

19 In one embodiment, an on-chip direct memory access
20 (DMA) controller controls the bus for configuring of the
21 memory cells. Using the DMA controller as a bus master
22 rather than the CPU reduces the number of clock cycles
23 needed for configuration, thereby reducing power
24 consumption. In another embodiment, the CPU controls the
25 bus during both configuration and read modes, thereby
26 reducing the number of on-chip components.

27 The present invention provides significant latitude to
28 the user to customize the configuration process. For
29 example, the user can select one clock during configuration
30 of the CSL and another clock after configuration of the
31 CSL. In one embodiment, the CPU initially sets key timing
32 parameters to facilitate accessing the slowest available

1 parallel memory device on the market and then accelerates
2 the access as appropriate for the specific memory device or
3 as selected by the user. Thus, the present invention
4 advantageously provides the user maximum flexibility in the
5 selection of the clock source, both in the configuration
6 mode as well as in the user mode.

7 In accordance with the present invention, the user can
8 also determine the initialization sequence of the CSoC.
9 For example, global buffers used for clock, high fan-out,
10 or reset signals can be selectively enabled in the sequence
11 designated by the user. Other elements of CSoC, including
12 but not limited to, registers, programmable inputs/outputs,
13 and RAM can be selectively enabled in the same manner.

14 The present invention provides certain transparent
15 features that provide optimal flexibility to the user. For
16 example, an initialization sequence first checks for either
17 a serial or a parallel external memory device coupled to a
18 memory interface unit (MIU) in the CSoC. The CPU sets the
19 MIU to an appropriate mode based on the type of external
20 memory device. Then, the CPU searches for a header in the
21 memory device. If the header is found, then the CSL is
22 configured. If the header is not found, the MIU is
23 switched to another mode. In one embodiment, if the header
24 is not found in the new mode, then the present invention
25 checks whether the memory device is accessible, powers down
26 the CSoC if the memory device is accessible, and repeats
27 the header search if the external memory device is not
28 accessible.

29 Another aspect of the present invention is providing
30 predetermined signals in the CSL until the completion of
31 configuration. Special buffers in the general interconnect
32 of the CSL, called zip buffers, provide this function.

1 Zip buffers include first-tier and second tier
2 multiplexers. Each first-tier multiplexer has input
3 terminals coupled to a set of general interconnect lines.
4 Each of the second-tier multiplexers has a first input
5 terminal coupled to one output terminal of the first-tier
6 multiplexers and a second input terminal coupled to a
7 constant voltage source. The second-tier multiplexers
8 selectively provide output signals from the first terminals
9 or the second terminals to the logic block. Specifically,
10 during the configuration mode, the second-tier multiplexers
11 provide only constant signals to the logic block, thereby
12 preventing contention within the logic block. During the
13 user mode, the second-tier multiplexers provide signals
14 from the general interconnect to the logic block.

15 The present invention also provides a unique input
16 multiplexer to receive the signals provided by the zip
17 buffers. The input multiplexer includes a first
18 multiplexer that receives a first plurality of input
19 signals and a constant signal, a second multiplexer that
20 receives a second plurality of input signals and a mode
21 signal, and a third multiplexer that receives output
22 signals from the first and second multiplexers and provides
23 an output signal on a logic block input line. The input
24 multiplexer further includes a transistor coupled to the
25 internal logic block line, wherein the first transistor is
26 controlled by the mode signal. In one embodiment, the mode
27 signal, which determines whether the logic block is in a
28 configuration mode or a user mode, is generated by control
29 logic in the CSL.

30 During configuration, the mode signal, the constant
31 signal, and all input signals are identical. To provide
32 further protection to the internal circuits of the CSL, the

1 mode signal also turns on the transistor coupled to the
2 internal logic block line. The transistor, which is
3 further coupled to the constant signal, thereby provides
4 this constant signal to the internal logic block line. In
5 this manner, the present invention ensures that the
6 internal logic of the CSL remains stable until
7 configuration is complete.

8 Because the configuration memory is randomly
9 accessible, just as any other memory on the CSoC, the
10 configuration memory can be addressed in any order, thereby
11 providing enhanced security and debug capabilities to the
12 user. Additionally, the configuration memory can be easily
13 addressed as subsets, thereby facilitating rapid
14 reconfiguration. In one embodiment, a user may choose
15 between various levels of security on the CSoC.

16 The present invention will be more fully understood in
17 light of the following description and drawings.

18
19 BRIEF DESCRIPTION OF THE DRAWINGS

20 Figure 1 illustrates a configurable system on a chip
21 (CSoC) in a typical system environment.

22 Figure 2 shows a block diagram of the basic structures
23 in a CSoC.

24 Figure 3, comprising Figures 3A, 3B, and 3C,
25 illustrates a flow chart of the configuration of a CSoC in
26 both a parallel and series mode.

27 Figure 4 shows a block diagram of the configurable
28 system logic including logic banks, horizontal breakers,
29 vertical breakers, corner breakers, and interface logic.

30 Figure 5 illustrates further details of a logic bank,
31 a horizontal breaker, and a vertical breaker.

1 Figure 6 shows a vertical breaker tile including
2 multi-purpose buses.

3 Figure 7A illustrates one embodiment of zip buffers
4 controlled by configuration memory cells.

5 Figure 7B shows one embodiment of an input multiplexer
6 controlled by configuration memory cells.

7
8 DETAILED DESCRIPTION OF THE DRAWINGS

9 A configurable system on a chip (CSoC) is a
10 monolithic, integrated circuit device that performs a
11 variety of microcontroller functions. Figure 1 illustrates
12 a CSoC 101 in a typical system environment 100. In this
13 environment, a memory device 103, after receiving the
14 appropriate chip enable (CE), write enable (WE), and output
15 enable (OE) signals from CSoC 101, provides CSoC 101 with
16 the configuration data for its programmable resources. In
17 one embodiment, memory device 103 is a flash read only
18 memory (ROM) which provides its output data in parallel.

19 Note that other CSoC system embodiments may use other
20 memory devices for configuration. These memory devices may
21 be implemented in other technologies, such as erasable
22 programmable read only memory (EPROM) or electrically
23 erasable read only memory (EEPROM) technologies. Moreover,
24 as described in more detail in reference to Figures 3A-3C,
25 the memory device may be a serial device instead of a
26 parallel device.

27 CSoC 101 is also connected to a JTAG connector 102.
28 JTAG, an acronym for the Joint Test Action Group, refers to
29 the IEEE Std 1149.1-1990 which defines a test access port
30 and boundary scan architecture for digital integrated
31 circuits. This standard is well known to those skilled in
32 the art and therefore is not described in detail herein.

Figure 2 illustrates the major structures of CSoC 101. CSoC 101 includes configurable system logic (CSL) 201, a programmable logic section to implement user-defined logic. CSL 201 is programmed by loading configuration memory cells (not shown). These configuration memory cells control the logic and general interconnect of CSL 201. A memory interface unit (MIU) 204 facilitates the transfer of the logic values for the configuration memory cells from an external memory (for example, memory device 103 (Figure 1)) to CSL 201 via a system bus 205. A random access memory (RAM) 202 provides user memory in addition to that provided by CSL 201. Programmable inputs/outputs (PIOs) 201A provide connection between CSL 201 and other resources (not shown) external to CSoC 101.

In accordance with the present invention, CSoC 101 advantageously uses an on-chip central processing unit (CPU) 206 during the configuration process. Specifically, CPU 206 executes a software program to write configuration data to the configuration memory cells of CSoC 101. Note that the configuration memory cells are found principally, but not exclusively, in CSL 201. Specifically, configuration memory cells are also included in PIOs 201A.

Figure 3, which comprises Figures 3A-3C, is a flow chart of the configuration process in CSoC 101. After power-on in step 301, MIU 204 (Figure 2) initializes as a multi-chip slave (step 302) that can be accessed by multiple external memory devices. If CSoC 101 is a slave chip, as indicated by a logic one signal on the SLAVE pin (not shown in Figure 1), CPU 206 is held in reset (step 303). If the signal on the SLAVE pin is a logic zero, thereby indicating that CSoC 101 is the master chip, then CPU 206 takes control and begins to execute the

1 initialization code stored in non-volatile memory 203 (step
2 304). In one embodiment, the initialization code is
3 divided between non-volatile memory 203 (internal) and
4 memory device 103 (external), wherein non-volatile memory
5 203 stores only a standard start-up sequence and memory
6 device 103 stores the user-specific code (described in
7 detail in reference to steps 318, 319 and 325).

8 In step 305, CPU 206 determines if CSoC 101 should
9 stay in a secure mode. In Figure 2, there are two points
10 of access to CSoC 101: JTAG connector 102 and memory device
11 103. Other environments may provide fewer or more points
12 of access. In accordance with the present invention, a
13 user may choose between various levels of security.
14 Specifically, in Figure 2, JTAG connector 102 or memory
15 device 103 may be selectively disconnected (individually or
16 in combination) from or remain connected to CSoC 101. This
17 security designation is translated into executable code and
18 stored in memory device 103. As indicated previously, CPU
19 206 may read a portion of memory device 103 in step 304.
20 In one embodiment, the user directs CPU 206 to deny access
21 to JTAG connector 102 as well as to external memory 103,
22 thereby preventing any readback of the user's design. If
23 memory device 103 is disconnected, then CSoC 101 must have
24 been previously programmed, i.e. all values already loaded
25 into the configuration memory cells, and a reset function
26 triggered. If CSoC 101 should be in a secure mode, CPU 206
27 executes the code stored in RAM 202 in step 305A.
28 Otherwise, CSoC 101 enters a non-secure mode 306.

29 A non-secure mode indicates that CSoC 101 will operate
30 in conjunction with external resources, such as memory
31 device 103. Therefore, in step 307, CPU 206 checks whether
32 memory device 103 is a parallel or serial device. As shown

1 in Figure 1, memory device 103 is a parallel device. Thus,
2 at this point, CSoC 101 would enter a parallel mode 308.
3 In step 309, MIU 204 is set in parallel mode.
4 Specifically, in one embodiment, CPU 206 initially sets key
5 timing parameters to facilitate accessing the slowest
6 available memory device on the market and then accelerates
7 the access as appropriate for the specific memory device.

8 In step 310, CPU 206 searches in memory device 103 for
9 a valid identification (ID) header. In accordance with the
10 present invention, multiple configuration files may be
11 stored in memory device 103. These files may be associated
12 with different devices or different functions performed by
13 CSoC 101. The ID header identifies a valid configuration
14 file in memory device 103.

15 If a header is not found in step 311, then CPU 206
16 determines whether a signal on the VSYS pin of CSoC 101 is
17 high in step 312. The VSYS pin is connected to the power
18 supply of devices external to CSoC 101, such as memory
19 device 103. If the power supply to CSoC 101 (pin VCC) is,
20 for example, 3.3 volts and the power supply to memory
21 device 103, is 5 volts, then memory device 103 might not be
22 fully powered up and thus not accessible at the time CPU
23 206 searches in memory device 103 for a valid ID header.
24 The present invention ensures that CPU 206 is given an
25 opportunity to search for the header when memory device 103
26 is accessible. Therefore, if the VSYS signal is low in
27 step 312, CPU 206 returns to non-secure mode 306 to begin
28 another read cycle.

29 On the other hand, if the VSYS signal is high, then
30 memory device 103 is accessible and CPU 206 should have
31 been able to find a valid ID header. Therefore, either a
32 valid ID header is not present or the data stored in memory

1 device 103 is corrupted. Either case requires user
2 intervention or correction before the configuration can
3 continue. Therefore, CSoC 101 is powered down in step 313.
4 Once powered down, CPU 206 waits until it receives a reset
5 (RST) signal from the user in step 314. At this point, CPU
6 206 returns to non-secure mode 306.

7 If a header is found in step 311, then CPU 206 begins
8 code execution (step 317) from memory device 103 at the
9 address indicated by the header. In one embodiment, the
10 header includes executable code to trigger this CPU
11 process.

12 After code execution begins, CPU 206 selects the user-
13 designated clock source in step 318. CSoC 101 may include
14 an internal ring oscillator (not shown), or may be coupled
15 to multiple external clock sources (such as a crystal
16 oscillator (Figure 1) via terminals XTAL1 and XTAL2).
17 Selection between multiple clock sources is generally
18 implemented with a multiplexer as shown by clock control
19 208 in Figure 2. This choice of clock sources provides
20 optimal flexibility in configuration speed to the user.

21 In step 319, CPU 206 executes a routine, if necessary,
22 which synchronizes external system resources (if present)
23 connected to system bus 205. Generally, the configuration
24 process requires significant data transfers. The present
25 invention advantageously provides a selectable interrupt or
26 wait command in step 319 before this data transfer occurs.
27 In one embodiment, CPU 206 waits for authorization from
28 these external system resources before continuing the
29 configuration process. For example, if external memory 103
30 is a dual-ported memory, then in step 319 CPU 206 may
31 receive an instruction to wait until that dual-ported
32 memory is fully loaded.

1 In step 320, CPU 206 sets up the parameters for a DMA
2 transfer, then DMA controller 207 takes over as the master
3 of system bus 205. Using DMA controller 207 as a bus
4 master rather than CPU 206 reduces the number of clock
5 cycles needed for configuration, thereby reducing power
6 consumption. At this point, DMA controller 207 uses system
7 bus 205 and MIU 204 to configure the internal programmable
8 resources of CSL 210 and PIOs 201A. Additional internal
9 programmable resources of CSoC 101, such as the control
10 register unit (CRU) which is distributed throughout CSoC
11 101, may also be programmed at this time. The CRU is the
12 repository of all control bits for semi-custom gates.
13 Thus, the sequencing that will be discussed in reference to
14 step 325, for example, is controlled by the CRU. Either
15 CPU 206 or DMA controller 207 programs the CRU.

16 Note that the present invention allows other bus
17 masters to configure CSoC 101. As noted previously, system
18 bus 205 is a multi-master bus. Therefore, in another
19 embodiment of the present invention, CPU 206 performs the
20 configuration of CSL 210 and PIOs 201A. In yet another
21 embodiments, an external device to CSoC 101, such as
22 another CSoC or a memory device containing executable code,
23 communicates with CPU 206 via MIU 204 and system bus 205,
24 becomes master of system bus 205, and continues the
25 configuration of CSoC 101. In yet a further embodiment, an
26 external device communicates with CPU 206 via JTAG
27 interface 102 and system bus 205, becomes master of system
28 bus 205, and continues the configuration of CSoC 101. For
29 purposes of illustration only, the following description
30 assumes that DMA controller 207 is bus master starting in
31 configuration step 320.

1 During the transfer of data from memory device 103 to
2 the above-mentioned internal resources, DMA controller 207
3 runs a verify CRC checksum algorithm in step 321 to improve
4 the integrity of those data signals. This CRC checksum
5 algorithm and the use of DMA controller 207 to run this
6 algorithm are well known in the art and therefore are not
7 explained in detail herein.

8 Assuming all data is transferred successfully, as
9 verified by DMA controller 207 in step 321, CPU 206 enters
10 the post CSL configuration setup stage 322. In this stage,
11 other user-specified data may be loaded into RAM 202 (step
12 323). This user-specified data may include values for
13 lookup tables (LUTs) and registers in CSL 201, specific
14 configuration memory locations in CSL 201, the CRU, or
15 programs to be run by CPU 201 in the user mode.

16 In one embodiment, the values written to RAM 202
17 include information for partial reconfiguration of the
18 programmable resources in CSL 201 during the user mode.
19 Because many programmable logic designs include repetitive
20 sections, this aspect of the present invention saves
21 considerable silicon real estate and time.

22 At this point, CSoC 101 is essentially programmed.
23 However, before entering the user mode, the clock source of
24 CSoC 101 may again be chosen as indicated in step 324.
25 This step, performed by CPU 206, allows the user to select
26 different clock sources for the configuration mode and the
27 user mode. For example, some users prefer a fast
28 configuration mode, but need a low power (i.e. slower) user
29 mode. Other users have no timing requirements during
30 configuration, but want to maximize system performance
31 during the user mode, for example by using high speed clock
32 generation circuitry including a phase locked loop. Thus,

1 the present invention advantageously provides the user
2 maximum flexibility in the selection of the clock source,
3 both in the configuration mode as well as in the user mode.

4 In accordance with the present invention, step 325
5 ensures proper system synchronization by allowing the user
6 to determine the enablement sequence of various elements of
7 CSoC 101. In one embodiment, the enablement sequence is
8 provided by the user during the capture of the user's logic
9 design and the appropriate instructions are generated by
10 the programming software and loaded into memory device 103.
11 Alternatively, the user may provide proprietary executable
12 code including the instructions for the enablement
13 sequence. In either case, these instructions are read by
14 CPU 206, which controls the synchronization process. The
15 following areas of selective enablement may be included in
16 step 325: CSL zip buffers and input multiplexers
17 (explained in detail in reference to Figures 7A and 7B),
18 global buffers (used for clock, high fan-out, or reset
19 signals), registers (i.e. flip-flops), and lookup table
20 (LUT) RAM in CSL 201.

21 CPU 206 also controls the selective enabling of PIOs
22 201A in step 325, thereby providing certain output signals
23 before other output signals from CSoC 101. A more detailed
24 description of PIOs 201A is provided in U.S. Serial No.
25 _____ (TRI-004), filed on October 15, 1999 by Triscend
26 Corporation, and entitled "An Input/Output Circuit With
27 User Programmable Functions", which is incorporated by
28 reference herein.

29 In one embodiment, the zip buffers are enabled first,
30 followed by the global buffers, the registers, and the PIOs
31 (a typical programmable logic device sequence) to ensure
32 that CSoC 101 drives the proper output signals to the

1 external system. In another embodiment, the registers and
2 PIOs of CSoC 101 are enabled and the system external to
3 CSoC 101 is allowed to stabilize before the zip and global
4 buffers are enabled.

5 Finally, also during step 325, CPU 206 allows for the
6 inclusion of customer, i.e. executable, code into the
7 sequence, thereby providing optimal flexibility to the
8 user.

9 In step 326, the user may selectively put CSoC 101
10 into a secure mode. In this manner, CSoC 101 will not read
11 from any external devices unless a power-down cycle is
12 triggered. In one embodiment, the secure mode allows
13 reading from only certain devices, such as memory device
14 103 or JTAG connector 102, or neither. In the secure mode,
15 if a reset signal is received, CPU 206 returns to step 305A
16 and executes the user code from RAM 202.

17 If a non-secure mode is chosen, then in step 327,
18 program control is transferred to the user code residing in
19 memory device 103 (note that even in a non-secure mode, CPU
20 206 can still access non-volatile memory 203). Accessing
21 memory device 103, although somewhat slower and higher in
22 power consumption because of toggling input/output
23 circuits, allows access to significantly larger memory
24 resources. The configuration mode ends in step 328,
25 thereby triggering the user mode.

26 As described in further detail below, the present
27 invention advantageously not only allows multiple memory
28 devices to be connected to CSoC 101, but also allows
29 various types of memory devices to be accessed, thereby
30 significantly increasing user flexibility. For example,
31 returning to step 307, if the external memory has a serial
32 interface, then a serial mode begins in step 330. In this

1 mode, CPU 206 sets MIU 204 in the serial mode (step 331),
2 resets the address pointer in the external memory device
3 (step 332), and then searches for a valid ID header in that
4 memory device (step 333). If a valid ID header is not
5 found, CPU 206 will assume another type of memory device is
6 connected to CSoc 101, and will automatically switch over
7 to the parallel mode 308. However, if the valid ID header
8 is found in the external memory device, then the serial
9 mode is continued in step 334. Specifically, the program
10 portion of the code is transferred in step 335 from the
11 external memory device to RAM 202 and executed by CPU 206
12 in step 336. At this point, CPU 206 continues the
13 configuration process as indicated in steps 318-328.

14 15 System Bus Resources

16 In accordance with the present invention, system bus
17 205 may be used for both configuration data transfer and
18 general interconnect. This dual use provides significant
19 silicon savings compared to conventional field programmable
20 logic devices that require dedicated wires on chip to
21 connect the external memory device to the configuration
22 memory cells.

23 Figure 4 illustrates one embodiment of CSL 201 in
24 which a plurality of logic banks 401 are arranged in rows
25 and columns. Relative to every logic bank 401, a vertical
26 breaker 402 is positioned on the top edge, a horizontal
27 breaker 403 is positioned on the left edge, and a corner
28 breaker 404 is positioned in the left corner. In this
29 embodiment, interface logic 405 distributes address signals
30 down every column defined by corner breakers 404 and
31 horizontal breakers 403.

Specifically referring to Figure 5, from each corner breaker 404, the address signals `sw_adr[31:0]` are distributed to the horizontal breaker 403 immediately below the corner breaker 404. These address signals are then distributed to the logic bank 401 to the right of horizontal breaker 403. Selection of the configuration memory cells within each logic block tile 501 is performed using well known circuitry and methods and therefore is not discussed in detail herein.

Note that each logic bank 401 comprises an array of logic block tiles 501, each horizontal breaker 403 comprises an array of horizontal breaker tiles 503, and each vertical breaker 402 comprises an array of vertical breaker tiles 502. Breaker tiles (both horizontal and vertical) include programmable connections between certain general interconnect lines (long lines) in adjacent banks, structures for distributing power, ground, and clock signals, as well as system-level circuitry to integrate CSL 201 with other components in CSoC 101, such as CPU 206.

Interface logic 405 (Figure 4) distributes write data signals `sw_dws` down every column defined by vertical breaker tiles 502 and logic bank tiles 501. In contrast, the read data signals `sr_drs` are propagated up through this same column to interface logic 405.

In accordance with the present invention, the buses for distributing the write data and read data signals can be used for both configuration memory and user data, thereby providing an extremely efficient and flexible resource on CSoC 101. Figure 6 illustrates one embodiment of a vertical breaker tile 502 including such dual-purpose buses. Specifically, bus 601 distributes the write data signals `sw_dws`. Registers 602 provide a registered version

1 of those signals (fw_dws) to the configuration memory cells
2 in the appropriate logic block tiles 501. In this
3 embodiment, the same data is provided to every logic block
4 tile 501 in the column of tiles, thereby providing
5 flexibility in the placement of resources connected to bus
6 601. Row select circuitry, well known to those skilled in
7 the art, is used to determine which of those logic block
8 tiles 501 should receive the configuration data.

9 In a similar manner, read data bus 611 can carry
10 signals from the general interconnect via bus 605 or from
11 the configuration memory cells via bus 604. Specifically,
12 multiplexers 606 selectively provide general interconnect
13 signals fr_drs or configuration memory signals n_lldata.
14 The selected signals are stored in registers 607, then
15 provided to multiplexers 608. Although not used if
16 configuration memory signals are selected, multiplexers 608
17 allow an arbitrary 4-way permutation of the assignment of
18 the read data signals to facilitate placement and routing
19 of user programmable logic. Multiplexers 609 receive the
20 output signals of multiplexers 608 as well as the signals
21 from registers 607 provided directly on bypass lines. If
22 the signals in registers 607 are configuration signals,
23 then multiplexers 609 are programmed to transfer the
24 signals on the bypass lines. The output signals of
25 multiplexers 609 are combined via OR gates 610 with the
26 corresponding read data signals from an adjacent logic
27 block tile in the same column, thereby forming bus 611.
28 Thus, depending on the signal selection by multiplexers
29 606, bus 611 can be used for general interconnect in CSL
30 201 or for configuration read lines.

31 Determining the states of the configuration memory
32 cells in logic block tiles 501 is highly advantageous in

1 debug operations. For a more detailed description of such
2 debug operations, see U.S. Serial No. _____ (TRI-002),
3 entitled "Bus Mastering Debugging System For Integrated
4 Circuits", filed by Triscend Corporation on October 15,
5 1999, and incorporated by reference herein.

6

7 Zip Buffers and Input Multiplexers

8 The present invention provides programmable zip
9 buffers and input multiplexers to the logic block tiles to
10 eliminate contention during configuration. Figure 7A
11 illustrates zip buffers 701A-701D having a two-tier
12 multiplexer structure in which signals from various lines
13 in the general interconnect (single length lines and/or
14 long lines) of CSL 201 are selectively chosen via
15 multiplexers 702 and provided to multiplexers 703. If a
16 logic one signal is provided to the control terminals of
17 multiplexers 703, the interconnect signals are buffered and
18 transferred to an input multiplexer 704 of a logic block
19 tile 705. Depending on the logic states of memory cells
20 706, a selected interconnect signal is buffered and then
21 provided on line 707.

22 In accordance with the present invention, CPU 206
23 ensures a logic zero Enzip signal is provided to the
24 control terminals of multiplexers 703 until step 325. In
25 this manner, only logic zeros are provided to all logic
26 block tiles 705 (via input lines 707), thereby ensuring a
27 clean start-up of CSL 201.

28 Figure 7B illustrates an embodiment of an input
29 multiplexer 704A for a logic block tile. In this
30 embodiment, two 5 to 1 multiplexers 710A and 710B each
31 receive four input signals in0-in3. This embodiment
32 assumes that all signals output from zip buffers 701

1 (Figure 7A) are inverted. Thus, all input signals in0-in3
2 to multiplexers 710 are logic ones. Multiplexer 710A has
3 one input terminal connected to voltage VCC, whereas
4 multiplexer 710B has one input terminal connected to
5 receive the inverted Enzip signal (EnzipN). Note that
6 memory cells 702A-702D selectively pass signals in0-in3,
7 respectively, whereas memory cell 702E selectively passes
8 signal in4.

9 During configuration a logic one signal is provided to
10 the gate of transistor 711 (Enzip = 0, EnzipN = 1), thereby
11 turning on that transistor. Because transistor 711 is
12 coupled to voltage VCC, a logic one signal is provided on
13 line 712, the input line to a logic block tile (discussed
14 in reference to Figure 5). Thus, irrespective of the
15 programming of memory cells 702A-702G, the signal on line
16 712 during configuration is constant (i.e. a logic one
17 signal). Thus, the present invention advantageously
18 eliminates any possibility of contention on line 712 to the
19 logic block tile. Additionally, the present invention
20 ensures that the internal logic of the CSL remains stable
21 until configuration is complete.

22 Note that after configuration, the EnzipN signal
23 mimics as a supply voltage, i.e. a constant logic zero
24 signal. Thus, in the user mode, the present invention
25 ensures a different, constant signal source for
26 multiplexers 710A and 710B.

28 Synopsis

29 As mentioned previously, prior art FPLDs have
30 dedicated wires to provide stored values to the
31 configuration memory cells. Moreover, the prior art FPLDs
32 provide dedicated hardware (i.e. shift registers) to load

1 those values into the configuration memory cells. Both of
2 these dedicated resources are used only during
3 configuration.

4 In contrast, the present invention provides an
5 extremely efficient use of system resources. As described
6 above, the data buses provided in CSL 201 (and forming part
7 of the system bus) can be multi-purpose, i.e. used for both
8 configuration and user logic in the general interconnect.
9 Moreover, the use of CPU 206 in the configuration process
10 eliminates the need for dedicated hardware for memory cell
11 loading.

12 As a further advantage of the present invention, all
13 configuration memory locations in CSL 201 and storage
14 locations in RAM 202 are mapped into the addressable memory
15 space of system bus 205. This mapping provides significant
16 advantages to the user. For example, any master of system
17 bus 205 can read from or write to specific locations,
18 thereby enhancing debug operations.

19 Thus, the present invention provides a silicon
20 efficient programmable logic solution while at the same
21 time providing users optimal flexibility in configuration
22 and system resources.

CLAIMS

1. A method of using a system bus on a configurable system on a chip (CSoC), the CSoC including configurable system logic (CSL), the method comprising:

configuring a memory cell in the CSL using the system bus; and

reading the memory cell in the CSL using the system bus.

2. The method of Claim 1 wherein the system bus is controlled by a direct memory access (DMA) controller during the step of configuring.

3. The method of Claim 1 wherein the system bus is controlled by a central processing unit (CPU) during the steps of configuring and reading.

4. A method of initializing a configurable system on a chip (CSoC), the CSoC including configurable system logic (CSL), the method comprising:

selecting one of a plurality of clocks before configuring the CSL;

configuring the CSL; and

selecting one of the plurality of clocks after configuring the CSL.

5. A method of initializing a configurable system on a chip (CSoC), the CSoC including configurable system logic (CSL), the method comprising:

1 checking for one of a serial and a parallel output
2 device external to the CSoC and coupled to a memory
3 interface unit (MIU) in the CSoC;
4 setting the MIU to an appropriate mode based on the
5 output device;
6 searching for a header in the output device;
7 switching to another mode if the header is not found;
8 and
9 configuring the CSL if the header is found.

10
11 6. The method of Claim 5, wherein if the header is not
12 found in the other mode, then further including:
13 checking whether the output device is accessible;
14 powering down the CSoC if the output device is
15 accessible; and
16 repeating the steps in Claim 5 if the output device is
17 not accessible.

18
19 7. A programmable interconnect structure providing a
20 signal to a logic block, the interconnect structure
21 comprising:
22 a plurality of interconnect lines divided into sets of
23 interconnect lines;
24 a plurality of first-tier multiplexers, each first-
25 tier multiplexer having input terminals coupled to a set of
26 interconnect lines and having an output terminal;
27 a plurality of second-tier multiplexers, each second-
28 tier having a first input terminal coupled to one output
29 terminal of the first-tier multiplexers and a second input
30 terminal coupled to a constant voltage source,

1 wherein the plurality of second-tier multiplexers
2 selectively provide output signals from the first terminals
3 or the second terminals to the logic block.

4
5 8. A method of providing signals to a logic block, the
6 method comprising:

7 selectively providing one of a plurality of
8 interconnect signals and a constant logic signal to the
9 logic block,

10 wherein during a configuration mode, the constant
11 logic signal is provided to the logic block, and

12 wherein during a user mode, the plurality of
13 interconnect signals are provided to the logic block.

14
15 9. An input multiplexer to a logic block, the input
16 multiplexer comprising:

17 a first multiplexer receiving a first plurality of
18 input signals and a constant signal;

19 a second multiplexer receiving a second plurality of
20 input signals and a mode signal;

21 a third multiplexer receiving output signals from the
22 first and second multiplexers and providing an output
23 signal on a logic block input line; and

24 a first transistor coupled to an internal logic block
25 line, the first transistor controlled by the mode signal.

26
27 10. A method of providing signals on an internal logic
28 block line of a logic block, the method comprising:

29 providing a plurality of input signals to a plurality
30 of selectors, wherein each selector further receives one of
31 a constant logic value and a mode signal;

1 selectively providing one of the plurality of input
2 signals, the constant logic value, and the mode signal on
3 the internal logic block line, wherein the mode signal
4 indicates whether the logic block is in a configuration
5 mode or a user mode.

6
7 11. The method of Claim 10, wherein during configuration,
8 the plurality of input signals, the constant logic value,
9 and the mode signal have the same value.

10
11 12. The method of Claim 10, wherein during the user mode,
12 the constant logic value and the mode signal have different
13 values.

14
15 13. The method of Claim 10, further including using the
16 mode signal to transfer the constant logic value to the
17 internal logic block line.

1 CONFIGURATION IN A CONFIGURABLE SYSTEM ON A CHIP

2 Brian Fox

3 Andreas Papaliolios

4
5 ABSTRACT OF THE INVENTION

6 The present invention allows a user to customize the
7 configuration sequence of a configurable system on a chip
8 (CSoC), thereby adding considerable flexibility to the
9 configuration process. The present invention also provides
10 certain features, transparent to the user, which optimize
11 system resources and ensure the correct initialization of
12 the CSoC. The CSoC leverages an on-chip central processing
13 unit (CPU) to control the configuration process of the
14 configurable system logic (CSL). Advantageously, the CSL
15 configuration memory cells as well as other programmable
16 locations in the CSoC are addressable as part of a system
17 bus address space. The system bus is a multi-use structure
18 that can be used for both configuring and reading of memory
19 cells. In this manner, the CSoC optimizes system
20 resources.

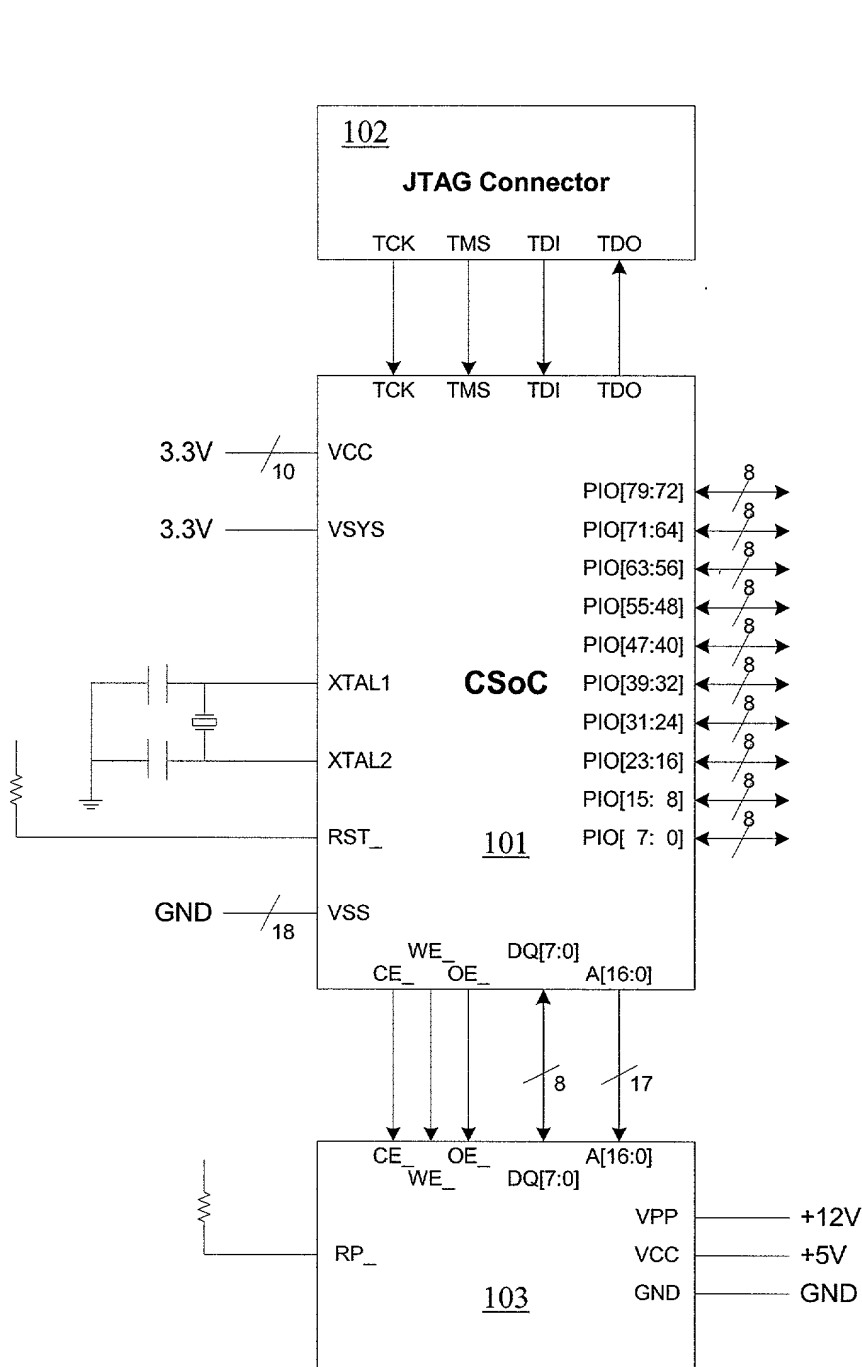


Figure 1

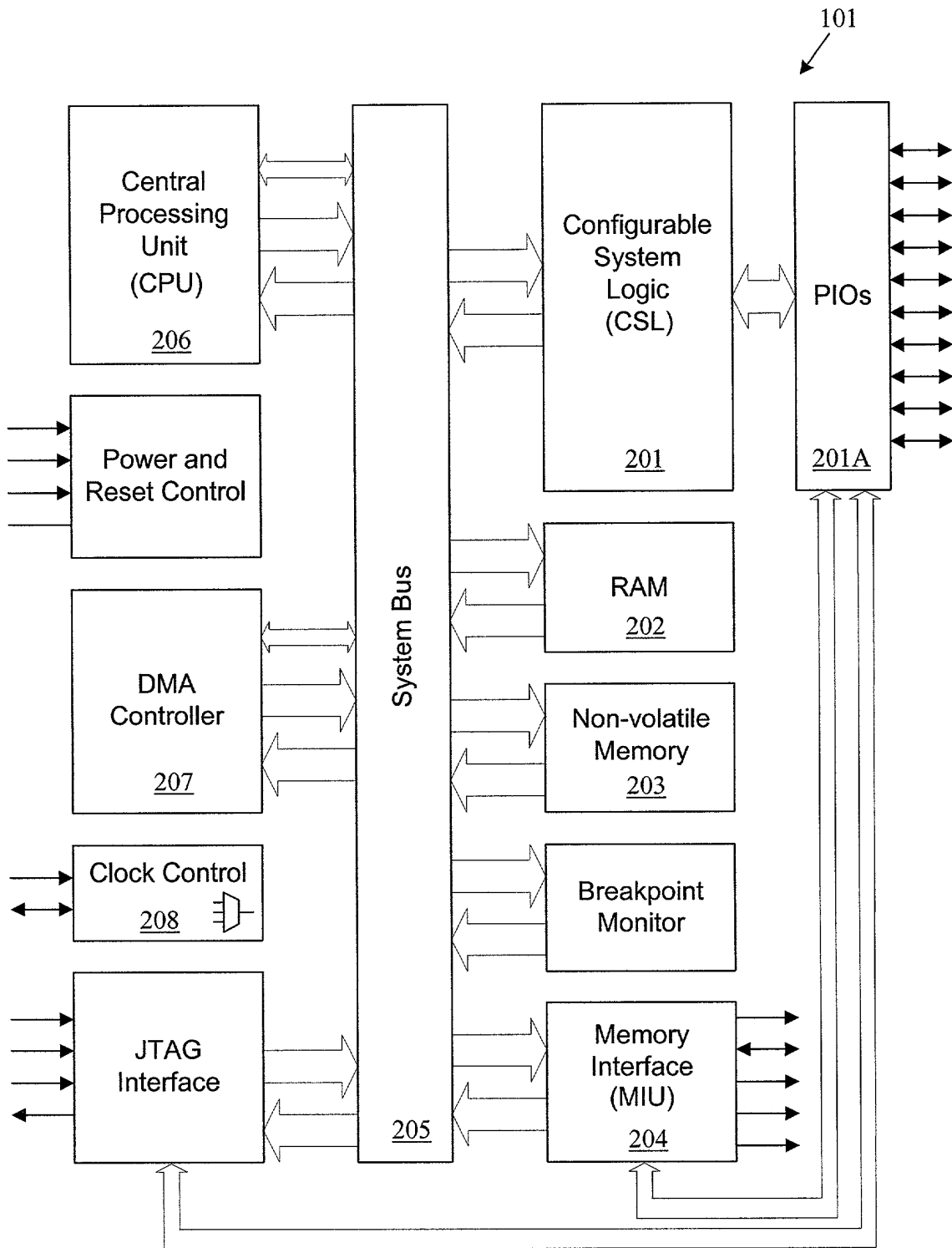


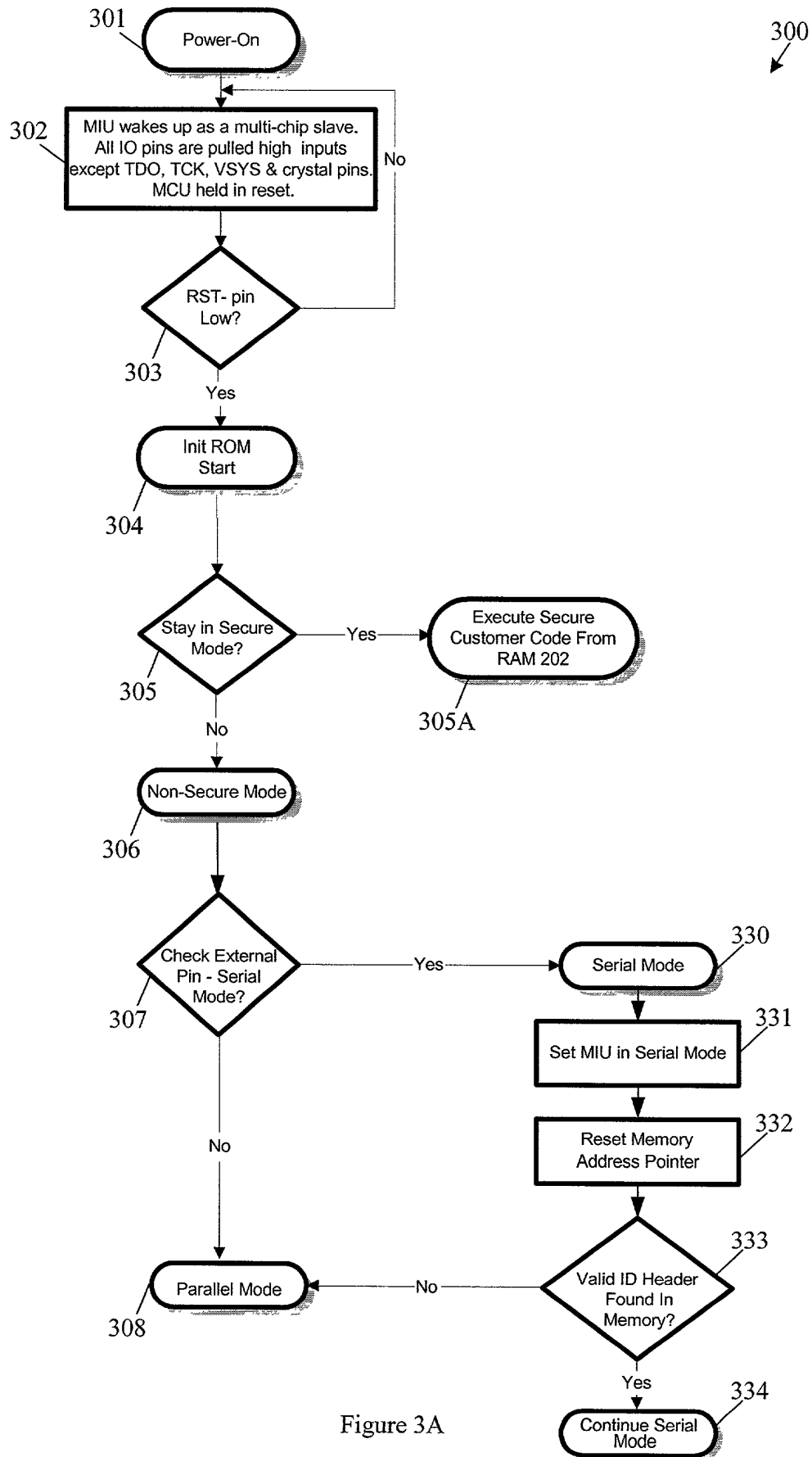
Figure 2

Figure 3A

Figure 3B

Figure 3C

Figure 3



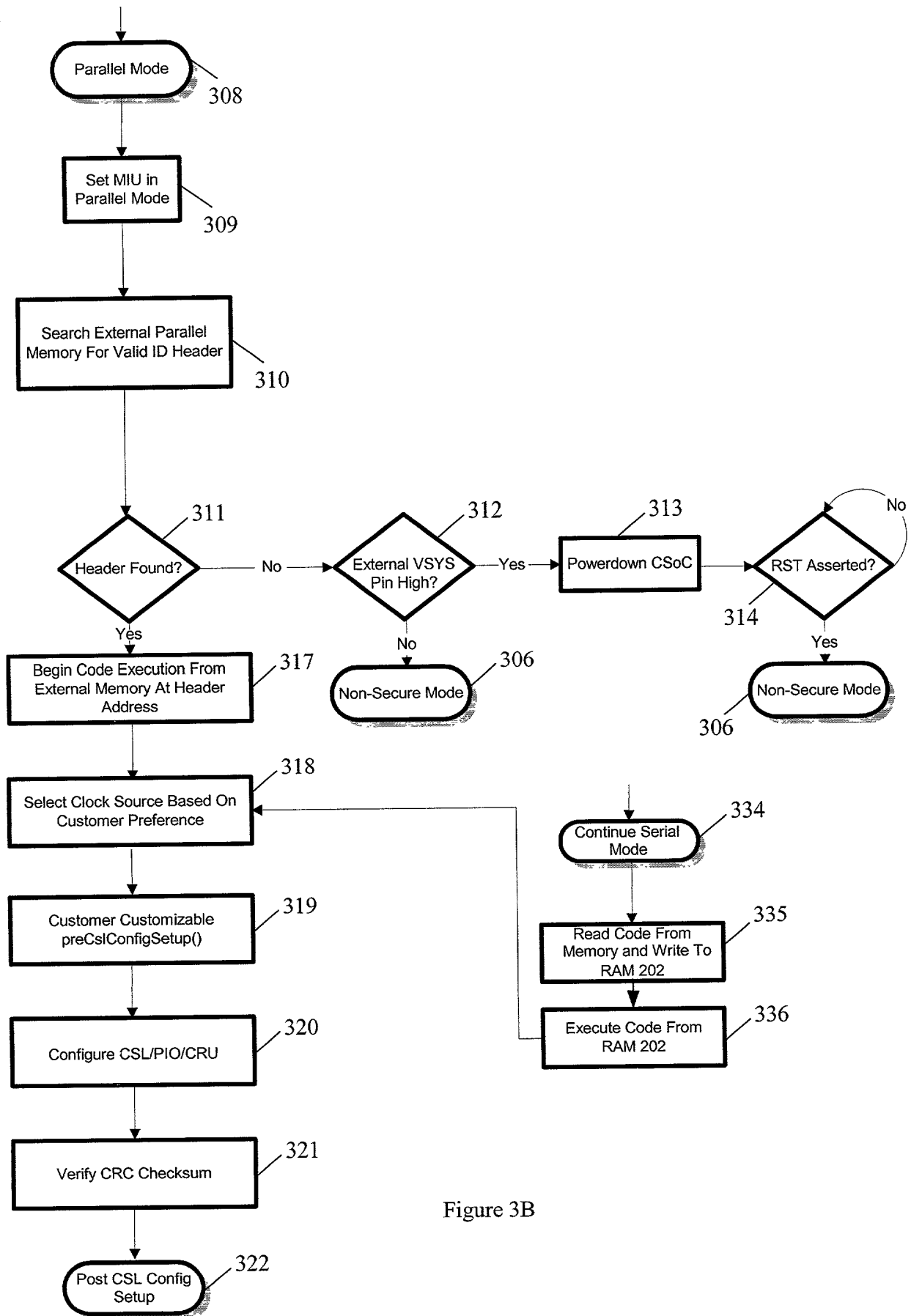


Figure 3B

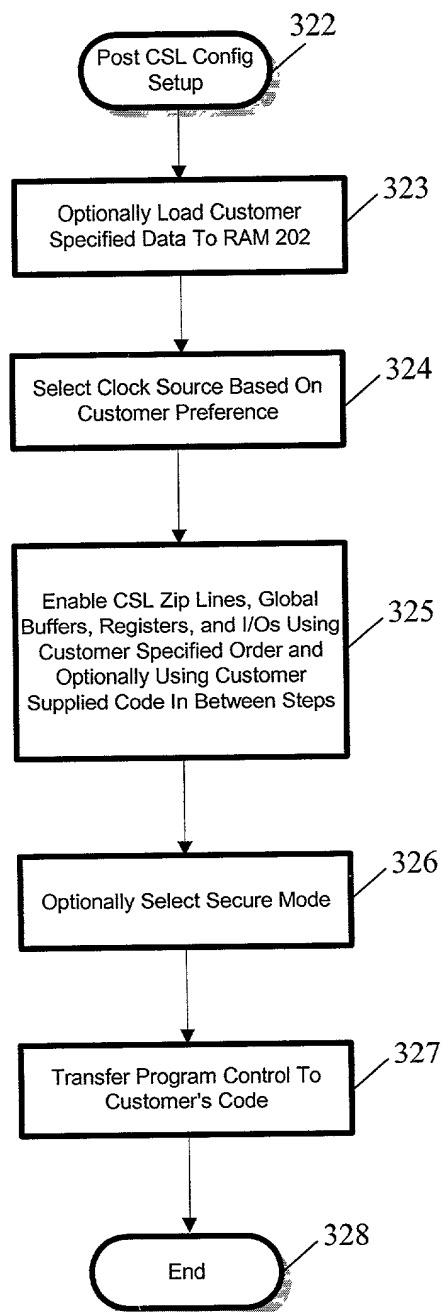


Figure 3C

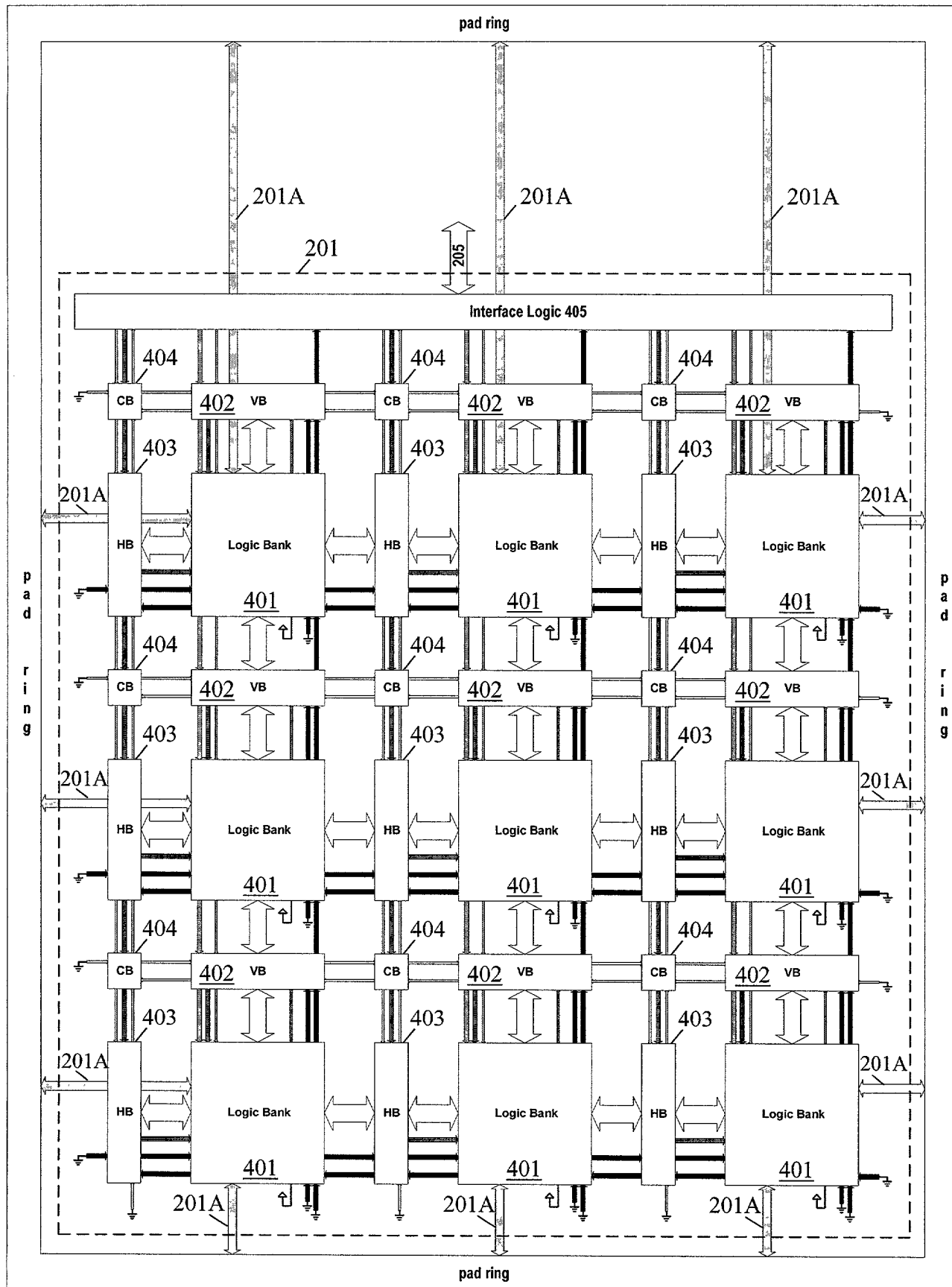


Figure 4

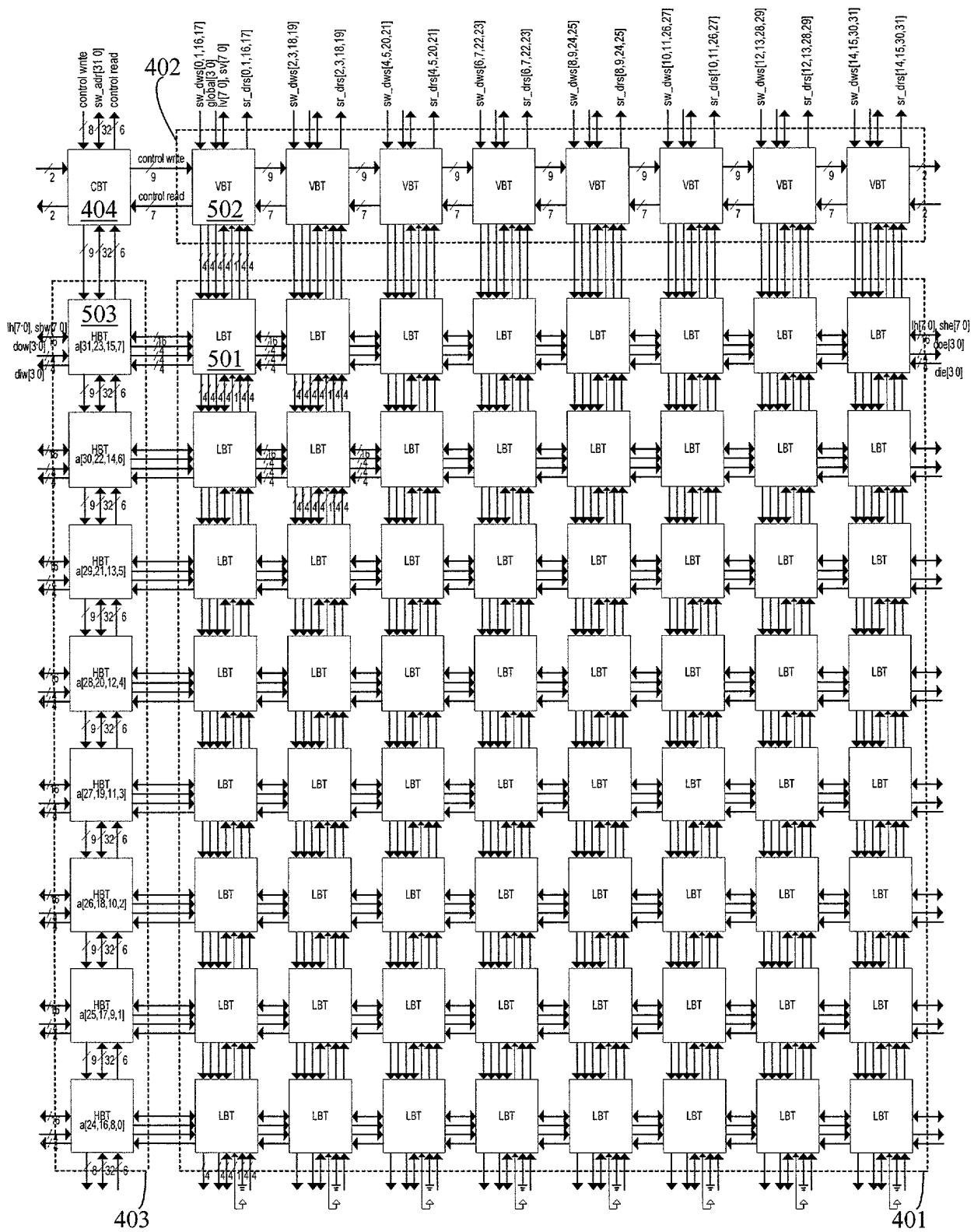


Figure 5

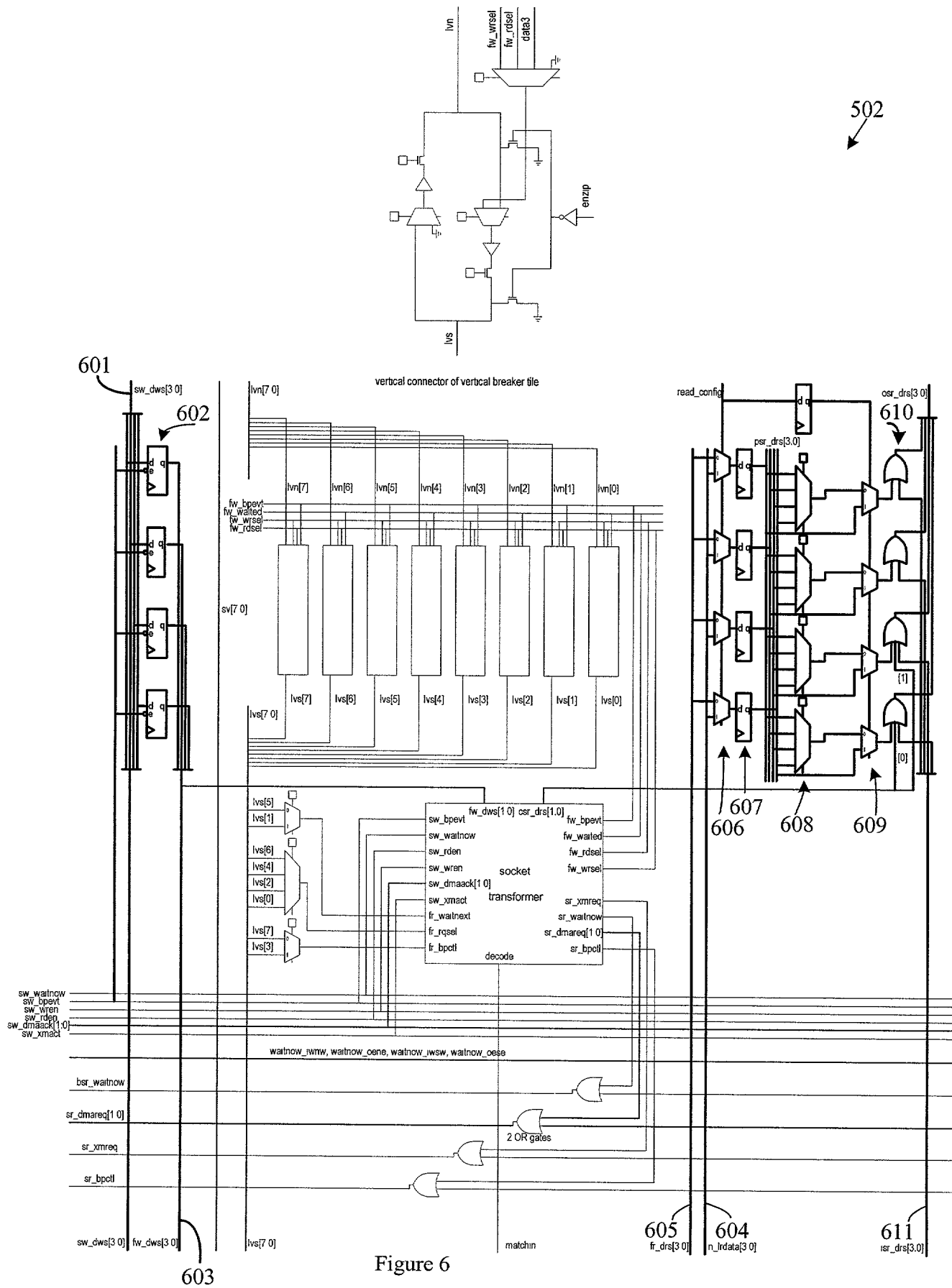


Figure 6

